Actas do Encontro Científico
3º Festival Nacional de Robótica- ROBOTICA2003
Lisboa, 9 de Maio de 2003

TARGET TRACKING USING FUZZY CONTROL

Nuno de Castro, Rodrigo Matias, M. Isabel Ribeiro

Institute for Systems and Robotics, Instituto Superior Técnico,
Av. Rovisco Pais 1, 1049-001 Lisboa, Portugal.
{nunocastro,rodrigo.matias@clix.pt}, mir@isr.ist.utl.pt

Abstract: This paper presents an approach to solve the target-tracking problem using a two-loop fuzzy logic controller, one for range control to the target and the other for orientation correction. These loops are assembled in a conversion block, whose output links general variables as linear and angular accelerations to the particular state-variables of the robot, resulting on a low-cost and easy implementable controller. The method was tested in simulation and in a real platform, using a differential-drive Nomad Scout. Both have achieved success in the task of following the target even when the system kinematics parameters and the sampling time suffer changes in their nominal values.

Keywords: fuzzy logic controller, target tracking, mobile robots, vision, differential topologies, Soccer Robotics, Nomad Scout.

1. INTRODUCTION

Target tracking is an important problem for a number of different applications that may benefit from the use of a mobile robot. Following an object at a given distance has application in several fields, namely in traffic control based on a «follow-the-leader» concept or in semi-automatic vehicles on an attempt to improve the road driving safety.

This paper addresses the problem of having a robot tracking a ball and following it at an arbitrary distance. The motivation to this particular application was the development of new solutions and behaviours to a Soccer Robot environment.

Every tracking procedure is divided in two main blocks whose roles are: seeking the object and following it. The first is an object detection problem, and is solved through the processing of sensor data. The second task is a pure control issue. This paper focuses essentially on the second problem, i.e., the controller design and implementation.

A classic controller can become highly complex when a system is non-linear or with complex dynamics. On the other hand, it is common to see human operators taking care of complex control tasks with good results and apparently without difficulty. They do not know the mathematical model of the system, and so the question that may be asked is on the approach they take to solve the problem. In an era where systems and its models get more and more complex, increasing the difficulties of the controllers design, simpler controllers, yet with similar performance, are always welcome.

When driving a car, the driver does not have a model for the distance to the front car, but has an idea of range like «near» or «far». These concepts are not binary-logic but fuzzy-logic instead. A simple set of rules working on range and orientation concepts can achieve the desired behaviour of the vehicle. This is the principle of a fuzzy controller addressing target tracking. In this paper, a fuzzy logic controller on range and angle is presented to follow a moving object at a certain distance.

The paper is organized as follows. Section 2 presents an overview of fuzzy logic and its control model. Section 3 reports the implementation done both in MatLab environment and in the real world using a Nomad Scout. The results are presented and discussed in Section 4, while the conclusions are drawn in Section 5.

2. OVERVIEW OF THE METHOD

A typical feedback control loop is represented in Figure 1. A fuzzy logic controlled system [Bezdek, 1993] uses the same control topology, even though the controller displays a special internal structure represented in Figure 2, with three blocks: a

fuzzification module, a set of *if-then* rules and a defuzzification unit [Custódio].

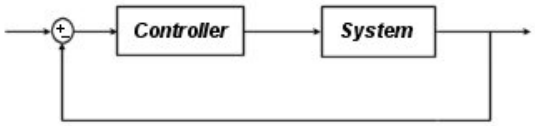

Fig. 1 – Typical control loop.

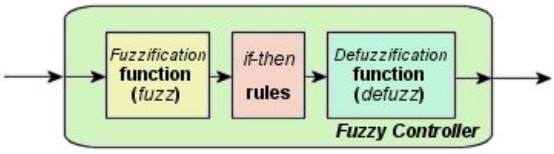

Fig. 2 – Internal structure of a fuzzy controller.

The fuzzification unit converts a quantified, numerical, control variable in a qualitative value like *small*, *medium* or *large*. These values cannot be considered as disjoint sets, once the frontier between each two is vague, fuzzy [von Altrock, 1995]. The Figure 3 describes this result, applying fuzzy sets [Zadeh, 1965] on a distance variable, as opposed to a division of the distance value in classical, disjoint sets.

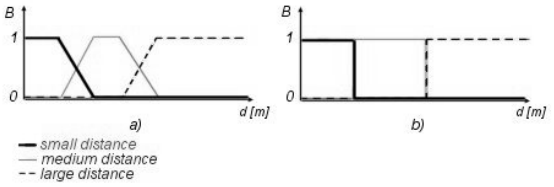

Fig. 3 – Fuzzy sets (a) in comparison to classic sets (b).

The block of *if-then* rules gathers the knowledge needed to successfully control the robot. With the fuzzified input variables, this unit uses the *Generalized Modus Ponens* rule to compute a qualitative output result for the controller [Zadeh, 1991]. This block plays a key role on the fuzzy controller operation as its rules model the whole behaviour of the vehicle.

The defuzzification unit converts a qualitative, fuzzy variable (e.g., *small, large)* in a quantitative, numerical value. There are many defuzzification methods. Among them, the centre-of-area method [Custódio; Cardoso, *et al.*, 1994; von Altrock, 1995] was chosen to solve the problem presented in this paper, since it has an easy-implementation and a fast execution time.

3. IMPLEMENTATION

3.1 Fuzzy Controller Design

To solve this particular issue of the target tracking problem a two-loop fuzzy controller was used, with one control loop for distance correction to the target and another loop for orientation correction. These loops do not influence each other's output: the variables related with the distance between the robot and the target produce a linear acceleration output through the range control loop (and only through it), while an angular acceleration results from (and only from) the processing on angle inputs by the orientation controller. This means that distance and orientation angle to an object are considered to be two separated phenomena when controlling a vehicle, which is not completely true in the real world. However, this is a good approximation given the envisaged soccer application.

Fig. 4 – The effect of following an object with same motion at different ranges on its orientation correction.

Figure 4 describes two situations of tracking an object that differ on the initial distance to the target. If the robot is correctly oriented to the target, the tracking control resumes to a distance control. This is the primary assumption made when implementing this controller: the distance control is done considering that the robot is always well oriented to the target, making the range correction independent of any angular variables. What happens when the orientation has also to be corrected? Figure 4 shows that, for a fixed object, the angle relative to the vehicle longitudinal axis decreases as the distance to it increases. When range d tends to infinity, the orientation angle correction, θ, tends to zero. However, similarly to a driver that steers only the sufficient to compensate the perceived error, regardless of the distance to the target, the orientation control loop can have the same behaviour, ignoring any range values to produce a valid angular acceleration output for the robot.

Linear and angular accelerations are the outputs of the two-loop fuzzy controller. These variables are general and independent of the particular characteristics of the vehicle under control. A conversion block makes the linkage of the general variables to the robot own state variables considering the limitations as motor saturation or limited values for acceleration. The implemented control system block diagram is represented in Figure 5.

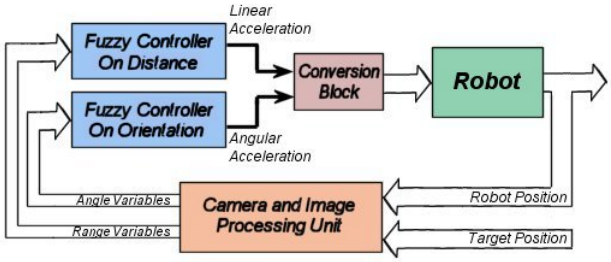

Fig. 5 – Block diagram of the target tracking fuzzy control system.

To compare the simulation results with those obtained with a Nomad, the *Robot* block has been modelled, using MatLab [The MathWorks, 2002], as a differential-drive vehicle, i.e., the platform motion is controlled from the angular velocities imposed on the right and the left wheels of the robot, as follows:

$$V = R/2\ (w_r + w_l) \quad (1)$$

$$W = R/D\ (w_r - w_l) \quad (2)$$

where V is the linear velocity of the robot, W the angular velocity of the robot, R the wheels radius of the vehicle, D the distance between wheels and w_r, w_l the angular velocities of right and left wheels.

Restrictions have been imposed on the values of w_r and w_l, namely a maximum value that leads to a saturation phenomenon for values above this maximum and a minimum value needed to move the robot. The robot model is completed with the inclusion of the motor dynamics of the two controlled wheels of the vehicle.

The *Conversion Block* uses (1) and (2) to compute w_r and w_l, the input state-variables of the robot, using the linear and angular accelerations that have resulted as output of the fuzzy controllers. However, (1) and (2) relate velocities instead of accelerations. Therefore, two integrators are needed at the conversion block input to convert the linear and angular accelerations into velocities.

The *Camera and Image Processing Unit* models a camera placed in the robot front and performs the image processing needed to extract, from acquired data, the distance and orientation between the vehicle and the target.

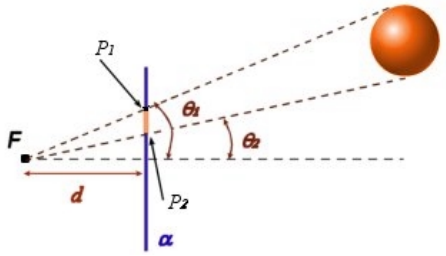

Fig. 6 – Image formation on the projective plane of a camera.

A representation on a projective plane has been used to model the camera, as represented in Figure 6. The camera is be characterized by a focus point, F, and a projective plane, α, where the image is formed. The focal distance d gives the range between F and α. The object position relative to the camera focus and its dimensions allow the estimation of the angles θ_1 and θ_2. The limit points of the object projection on α, P_1 and P_2, are then given by

$$P_1 = d\ tg(\theta_1) \quad (3),$$
$$P_2 = d\ tg(\theta_2) \quad (4).$$

To account for uncertainty, the camera model adds 10% noise to its output. Moreover, a delay element that represents the time needed by the camera to acquire an image and store it in memory is considered.

The second part of this unit is the image processing component. Knowing the focal distance d of the camera (which generally means that the camera has to be previously calibrated), the algorithm uses (3) and (4) to obtain θ_1 and θ_2. The target dimensions are then used in association with the computed angles to determine the object distance to the focus. The orientation of the target relative to the camera can be obtained using θ_1 and θ_2.

The fuzzy controller blocks presented in Figure 5 have the same internal structure as the one in Fig. 2. The fuzzification unit of the controller *On Distance* has two inputs, the range value to the target and the range change rate.

The fuzzification of a variable can result from different belonging functions, with different shapes and therefore different mathematical models. Fig. 7 gives examples of functions that can be used in a fuzzification procedure.

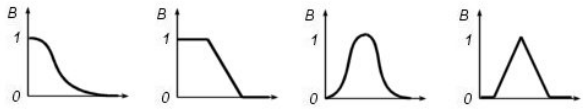

Fig. 7 – Different fuzzification functions.

Although there are many possible shapes for a belonging level function, B, the form of the particular function used does not influence in a significant way the output result of a fuzzy controller. Therefore, in this work, the fuzzification of the variables uses linear functions, aiming at an easy implementation and fast execution.

An important issue is the number of qualitative levels that should be used to fuzzify an input variable. A large number of fuzzy levels decreases the controller performance and increases the complexity of the implementation, while a small numbers renders the controller inefficient. The ideal number of levels generally sits between 3 and 5. In this work we used the three-level decomposition on variables that are strictly positive, like a range value, and a five-level decomposition of variables that could be assigned to both positive and negative values.

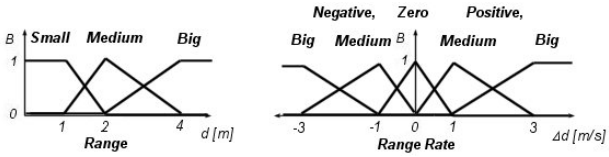

Fig. 8 – Fuzzification of range and range change rate values.

Figure 8 represents the decomposition in fuzzy levels of the *Fuzzy Controller On Distance* inputs. The division of the variables in 3 levels for distance and 5 levels for distance change rate leads to a set of 15 *if-then* rules. These rules, shown on Table 1, were obtained from the knowledge of a driver who wants to keep a medium distance to an object moving in front of him. In this table Pos. stands for Positive, Neg. for Negative and Med. for Medium.

Table 1 – Implemented if-then rules for distance control to the target.

	Distance		Distance C. Rate		Linear Acceleration
If	Small	&	Neg., Big	*Then*	Neg, Big
	Small		Neg., Med		Neg, Big
	Small		Zero		Neg, Med
	Small		Pos, Med		Neg, Med
	Small		Pos, Big		Zero
	Med		Neg, Big		Neg, Big
	Med		Neg, Med		Neg, Med
	Med		Zero		Zero

Med	Pos, Me	Pos, Med
Med	Pos, Big	Pos, Big
Big	Neg, Big	Zero
Big	Neg, Med	Pos, Med
Big	Zero	Pos, Med
Big	Pos, Med	Pos, Big
Big	Pos, Big	Pos, Big

The *if-then* rules block computes a qualitative value for the linear acceleration applying the *Generalized Modus Ponens* to the above rules. The defuzzification block converts this qualitative value in a numerical value, using the centre-of-area method. The *Generalized Modus Ponens* rule returns a value in the interval [0, 1] for each qualitative level of the linear acceleration as illustrated in Figure 9.

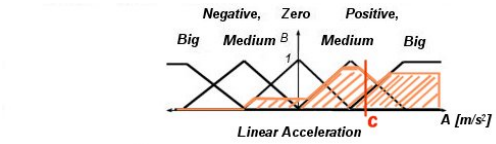

Fig. 9 – Defuzzification using the centre-of-area method.

The defuzzification operation finds the centre-of-area, *C*, obtained applying the values given by the *Generalized Modus Ponens* rule to each qualitative level.

The inputs of the *Fuzzy Controller On Orientation* block are the orientation angle between the robot and the target and the orientation angle change rate. The fuzzification method is the same used in the controller on distance, but with 5 qualitative levels for each variable. Figure 10 shows the division in qualitative levels made for both angle and angle change rate values.

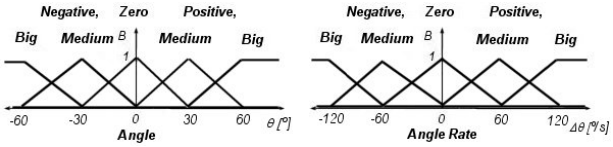

Fig. 10 – Fuzzification of angle and angle change rate values.

With 5 qualitative levels for each variable, 25 *if-then* rules are needed to cover all the possible combination of variables. The rules in Table 2 were designed based on common-sense criteria.

Table 2 – Implemented *if-then* rules for orientation correction of the robot.

	Angle		Angle C. Rate		Angular Acceleration
If	Neg, Big	*&*	Neg, Big	*Then*	Neg, Big
	Neg, Big		Neg,Med		Neg, Big
	Neg, Big		Zero		Neg, Big
	Neg, Big		Pos, Med		Neg, Med
	Neg, Big		Pos, Big		Zero
	Neg,Med		Neg, Big		Neg, Big
	Neg,Med		Neg,Med		Neg, Big
	Neg,Med		Zero		Neg, Med
	Neg,Med		Pos, Med		Zero
	Neg,Med		Pos, Big		Pos, Med
	Zero		Neg, Big		Neg, Big
	Zero		Neg,Med		Neg, Med
	Zero		Zero		Zero
	Zero		Pos, Med		Pos, Med
	Zero		Pos, Big		Pos, Big
	Pos, Med		Neg, Big		Neg, Med
	Pos, Med		Neg,Med		Zero
	Pos, Med		Zero		Pos, Med
	Pos, Med		Pos, Med		Pos, Big
	Pos, Med		Pos, Big		Pos, Big
	Pos, Big		Neg, Big		Zero
	Pos, Big		Neg,Med		Pos, Med
	Pos, Big		Zero		Pos, Big
	Pos, Big		Pos, Med		Pos, Big
	Pos, Big		Pos, Big		Pos, Big

As on the controller on distance, this *if-then* rules block returns a qualitative output after applying the *Generalized Modus Ponens* to the rules. The qualitative value of the angular acceleration is then converted in a numerical value using the same defuzzification method that was applied on the controller on distance, i.e., the centre-of-area method, but changing their boundary values to produce an angular acceleration compatible with the linear acceleration.

The rules implemented both on Table 1 and Table 2 are based on non completely objective criteria, this meaning that the achieved solution is not unique. Other fuzzy controller designers can achieve success in the target tracking task with different sets of rules. Nevertheless, these sets will never be significantly different from the ones presented in this paper.

3.2 Nomad Scout Following a Ball

The implementation on Nomad Scout [Nomadic Technologies, 1997] was done under a Soccer Robot environment, using the tools designed by the IST Soccer Robotics team, SocRob. These tools perform an automatic image data processing, returning the distance and the angle between the robot and the ball. The controller was implemented using the *C/C++* environment, whose similarity with the MatLab programming previously used for simulation purposes, simplified its implementation.

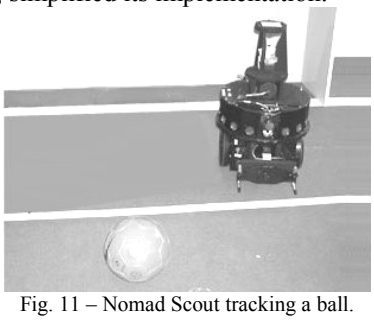

Fig. 11 – Nomad Scout tracking a ball.

Figure 11 shows the Nomad Scout that was used on this work. It has a camera with omni-directional mirror and a unidirectional camera oriented to the robot front. The first camera has the advantage of being able to see the target wherever it is. However, it has low precision and requires a complex, and thus slower, image processing when compared with a unidirectional camera.

The front camera is fast and accurate after being adjusted. It has, however, the drawback of being only able to see objects in front of the robot. Both cameras

have been used, in an attempt to avoid each camera limitations. In a first stage, the front camera is used, since its accuracy and speed are desirable to have a good performance of the system. If the target object is not detected using the front camera, the system tries to find it using the slower, inaccurate, omni-directional camera.

This procedure improves the robot task of tracking an object, since it combines the best characteristics of both cameras, resulting on a vision system with good accuracy and fewer gaps of «blindness» of the robot relative to the object.

4. EXPERIMENTS

Several experiments have been conducted to test different aspects of the implemented control system. Subsections 4.1 and 4.2 report MatLab simulation results and results obtained with the Nomad robot, respectively. On the simulation results, the controller was tested when tracking steady and moving targets. The controller robustness was also tested on parameter change of the system model, namely its kinematics constraints, the sampling time used and the control loop delay.

The tests done with Nomad aimed, essentially, at providing a confirmation of the theoretical results achieved at the simulation level.

4.1 Simulation Results

All the simulation tests consider a tracking distance to the target of 2 meters and a tracking angle to the target of zero degrees, aiming at having the robot following an object and always oriented to it. All experiments, except those concerning the effects of the sampling time change on system performance, used a sampling time of 100ms.

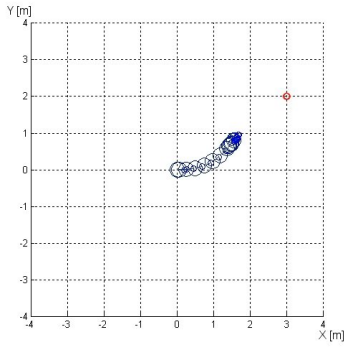

Fig. 12-a) Behaviour of the ideal robot facing the steady reference (3, 2) [m].

The first experiment obtained the controller step response for ideal conditions (no noise, null-delay of the camera and ideal motor response). This corresponds to having the robot following a steady target located at a distance and orientation to the vehicle different from their default values. Figure 12-a) describes the movement of the robot, while Figure12-b) gives the time-related range and angle correction of it (and their change rates).

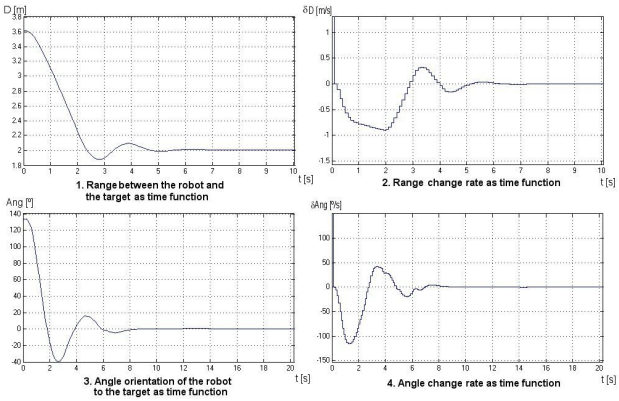

Fig. 12-b) Time response of the ideal robot on range and angle correction facing the steady target.

As it can be seen, the ideal system is well controlled by the implemented fuzzy controller. The system behaviour is underdumped on both control loops, which was expected since the controller is based on human reaction.

The same experiment was performed with a model nearer to real, i.e., with the inclusion of motor saturation at *1m/s*, a motor deadzone for velocities under *2cm/s* and camera noise of *10%* added to an output delay of *50ms*. This camera delay corresponds to the time interval required for the image acquisition and storage in memory. The results are shown in Figures 13-a) and 13-b).

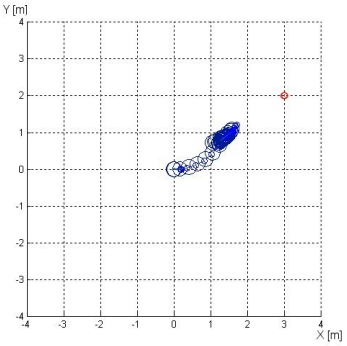

Fig. 13-a) Behaviour of the robot facing the steady reference (3, 2) [m], with 50ms of camera delay, noise level of 10%, motor saturation at 1m/s and motor deadzone below 2cm/s.

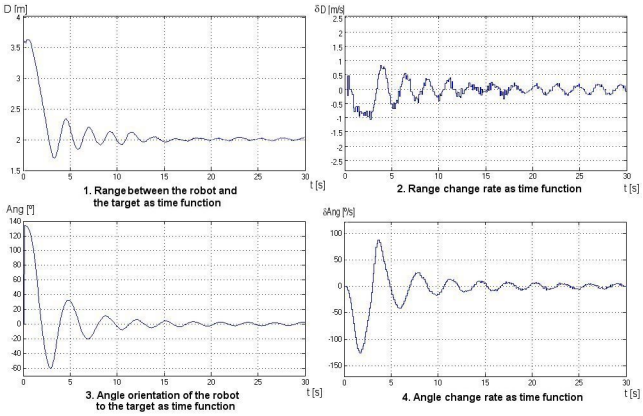

Fig. 13-b) Time response of the robot on range and angle correction facing the steady target.

The main difference between the two experiments is the longer time interval necessary to stabilize the robot posture. This is due to the camera delay and its noise. However, this kind of behaviour is not

particular to the fuzzy controller. Noise and delay on the control loop affects all controller types (classic and non classic), in a similar way.

When tracking non-steady targets, the fuzzy controller kept a good performance. An example is presented in Figures 14-a) and 14-b), which has been simulated with the same system parameters as the experiment above (*delay = 50ms, noise level of 10%, saturation at 1m/s and dead-zone below 2cm/s*) and a target (red) moving at a medium velocity of 20cm/s and a maximum of 50cm/s:

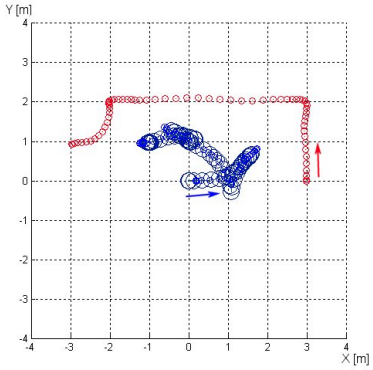

Fig. 14-a) Behaviour of the robot (blue) facing an arbitrary trajectory of the target (red).

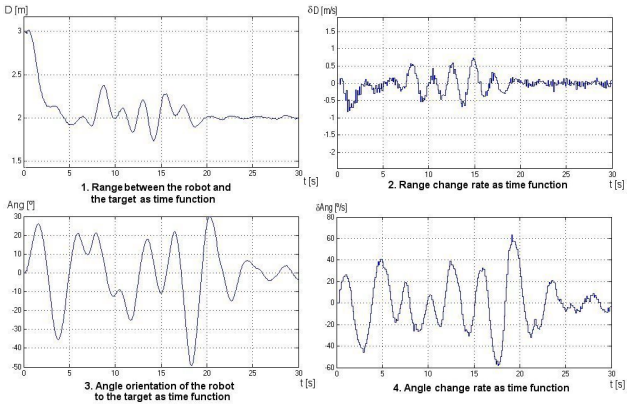

Fig. 14-b) Time response of the robot on range and angle correction facing the target trajectory.

The robot was able to track and follow the target, even when the last one suffers abrupt changes of direction, as the 90 degree turns represented in Figure 14-a). The system limitations, namely the motor saturation, motor deadzone, camera delay and noise, have not influenced in a significant way the success of the target following task. The controller was tested with many other target trajectories having similar results, this supporting the idea that the fuzzy controller has good performance on a system with the parameters and limitations considered.

When following a target at a certain distance, different from zero, the robot and the target trajectories are not necessarily the same. This was already seen in the previous experiment, but the best way of displaying this behaviour is the example given in Figure 15.

Following a target that moves in a circumference centred on the robot with radius equal to the default following distance makes the vehicle spin around itself, instead of moving on the circumference after the target, at 2 meters of it. However, the robot is always at the default target distance, keeping intact the property of following the target object.

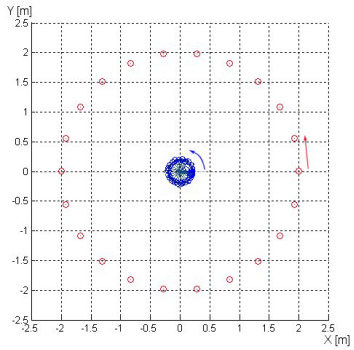

Fig. 15 – Different robot and target trajectories maintaining the property of following the target.

Another result to be achieved is the robustness of the implemented controller on parameter change. Maintaining the whole model unchanged, the value of the wheels radius and the distance between wheels was modified by 10%. To maximize the variation induced on the system by these parameters, it was added 10% to the wheels radius and 10% was subtracted to the distance between wheels. The results, following the same target of Figure 14-a), are shown in Figures 16-a) and 16-b):

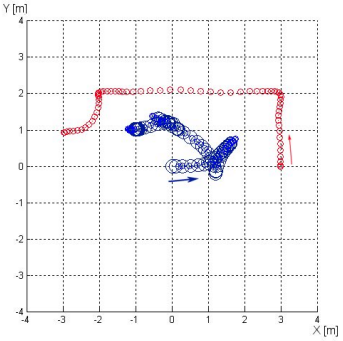

Fig. 16-a) Behaviour of the robot (blue) facing a target trajectory (red), having 10% change on its kinematics parameters.

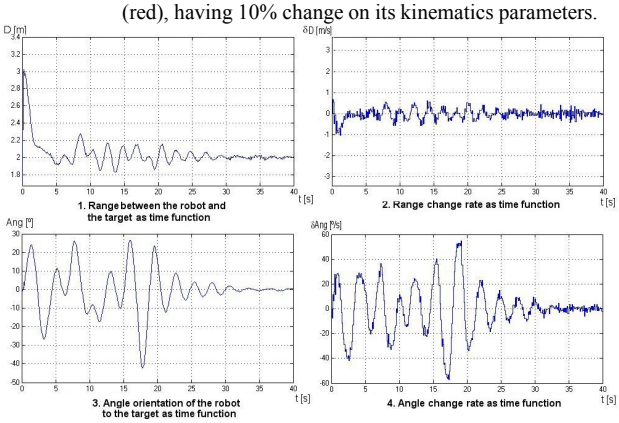

Fig. 16-b) Time response of the robot on range and angle correction facing the target trajectory.

Comparing Figures 16-a) and 14-a), it can be seen that the behaviour of the robot in the two situations is almost the same. The results of Figure 16-b) show that the controller still manages to drive the robot after the target, having however a slightly longer delay on point stabilization at the end of the robot

trajectory than that in Figure 14-b). Nevertheless, the example shown above supports the idea that the implemented fuzzy controller is robust to parameter variation as great as 10%, which can be considered a good result, since it gives a range of tolerance in robot measurements of centimetres in some cases.

The robustness of the controller has also been tested when changing the sampling time of the system and its camera delay value. Figures 17-a) and 17-b) reflect the results obtained with a camera delay and a sampling time 20% greater than the original ones (i.e., a camera delay of *60ms* and a sampling time of *120ms*):

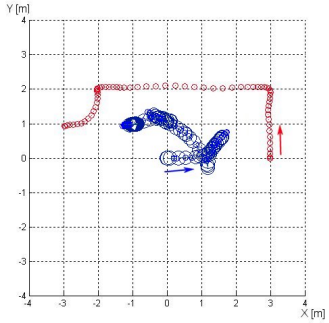

Fig. 17-a) Behaviour of the robot (blue) facing a target trajectory (red), with a sampling time of 120ms and a camera delay of 60ms.

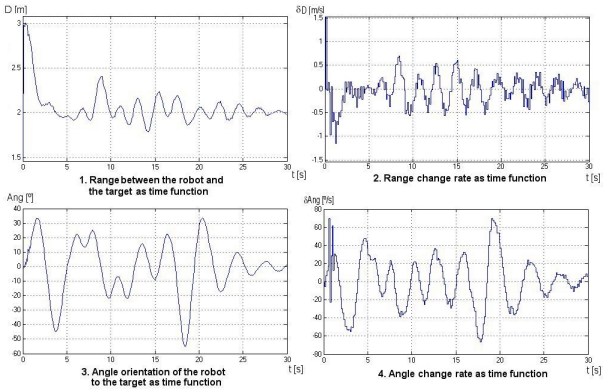

Fig. 17-b) Time response of the robot on range and angle correction.

Following the same trajectory as shown in Figures 14-a) and 16-a), the robot successfully performs its task, tracking the target with minimal differences on the trajectory made when compared with the results of 14-a) and 16-a). It is only in Figure 17-b) that is shown the main difference of the robot behaviour when compared with the previous ones: the existence of a bigger oscillation on the values of distance and angle between the robot and the target during its control.

This effect gets worse when the sampling time and the camera delay values increase. As in any classical control system, these parameters are critical for system stability and performance. To evaluate how much these parameters could influence the operation of the controller, a last experiment was conducted, simulating the system with the sampling time and the camera delay values doubled. The results are displayed in Figures 18-a) and 18-b), changing the sampling time of the system to 200ms and the camera delay to 100ms.

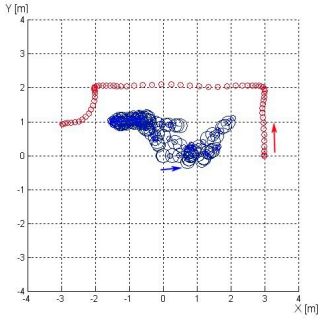

Fig. 18-a) Behaviour of the robot (blue) facing a target trajectory (red), with a sampling time of 200ms and a camera delay of 100ms.

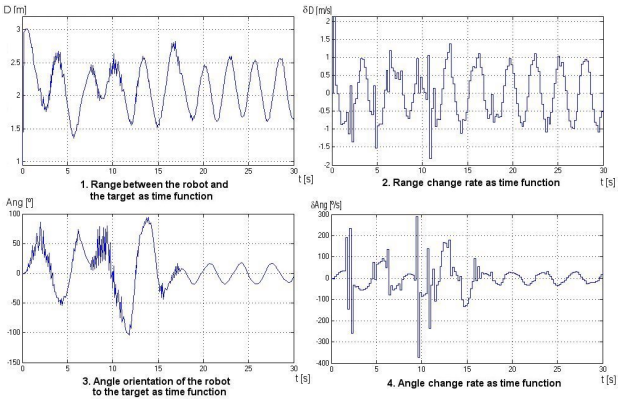

Fig. 18-b) Time response of the robot on range and angle correction.

As seen in Figure 18-a), the resulting movement of the robot is completely different from the previous ones, being much more inaccurate. In Figure 18-a), the vehicle does not stabilize at the end of the target trajectory, staying in an oscillation state instead. This behaviour is confirmed by Figure 18-b), where both distance and angle from the robot to the target have a non-attenuated oscillatory behaviour. The sampling time and the camera delay values chosen in this experiment bring the system to a critical state of stability, very close to instability.

When simulating the system with values above the 200ms for sampling time and 100ms for camera delay, the robot loses the ability to successfully track a target, and the system becomes unstable.

4.2 Results obtained with Nomad

The implemented controller is tested on a real robot. A Nomad Scout was used to track and follow an orange football ball. When testing the fuzzy controller with the Nomad, some difficulties have been experienced, which essentially result from the large sampling time used by the robot. The image processing proved to be very time-consuming compared with the control algorithm, pushing up the sampling time to values of about 200ms, which is not suitable to track successfully quick targets like a moving ball in a football game, as seen in the simulation results, Figures 18-a) and 18-b).

Furthermore, the image processing algorithm revealed a high-sensibility to light conditions, this yielding gaps of «blindness» of the robot when

following the target, making him stop until the ball is seen again.

With good light conditions and a slow moving target, the fuzzy controller showed good performance, similar to that obtained in the simulated environment. When tracking a still ball (steady-state behaviour), it was seen that the robot did not stay still after reaching the desired range and angle to the target, having small movements around these values instead. This was due to the camera noise, which caused a reasonable variation in the measurement of range and angle, and the sampling time. Nevertheless, given the robot limitations, it can be considered that the fuzzy controller had fine performance and proved to be robust both in simulation and in real operation, accomplishing the mission for which it was designed. The experiments made on Nomad are documented on video and available to anyone that requests them to the authors.

5. CONCLUSIONS

The results obtained prove that there is an effective alternative to a classical controller on tasks like target tracking by a mobile robot. A simple controller, based on a set of few *if-then* rules and using the fuzzy logic concept can take care of systems whose complexity or lack of model linearity could bring a big headache to a classical controller designer. However, this method has to be used wisely, once there are no stability criteria to apply on a fuzzy controller. The rules and the fuzz/defuzz functions usually match the common sense or the knowledge of an operator, so caution is required when implementing these parts of the controller to correctly reproduce the operator's actions, or else the behaviour of the controlled system can be completely unexpected.

As usual, the sampling time of the system is determinant to its performance. This is a key issue when working with Nomad, whose sampling time is about 200ms, a value that starts to be huge to get good results on tracking quick targets. The camera delay was another conditioning factor of the system performance.

The fuzzy controller proved to be robust even when the physical system parameters differ from the original ones.

A great advantage of the designed fuzzy controller is the fact of being completely independent of the system topology. The controller outlets are the linear and angular accelerations of the mobile robot, which are independent of the system is being controlled. The link between the controller and the system is the conversion block that transforms the controller outputs into the input variables of the robot. So, to control in the same way a robot that has a different kinematics, the only modification to be done is the conversion block instead of changing the whole controller. This has the additional advantage of making possible the inclusion of the non-linear limitations of the vehicle, like motor saturation, directly on the conversion block, maximizing the controller performance to each particular robot.

The partitioning of the controller in two control loops, one for distance control, the other for angle correction, simplifies the type and the number of rules, treating each one of them independently and only linking those in the conversion block.

In future developments it is suitable to improve the range and angle estimation given by image processing, as also decreasing the sampling time of the system and delay of the camera values. Except for the last problem, which can only be improved by hardware upgrade, the other improvements can be achieved increasing the quality of the image segmentation algorithm and its speed. These developments will improve the system performance, allowing the tracking of faster targets with smaller oscillation and increasing the stability margin of the system.

ACKNOWLEDGMENTS

The authors of this paper want to thank the whole ISocRob Project team, in particular Pedro Pinheiro and Hugo Costelha.

Work supported by the FCT "Programa Operacional Sociedade de Informação (POSI)" in the frame of QCA III.

REFERENCES

Zadeh, L., *Fuzzy Sets*, In Inf. Control, 8, 338-353, 1965.

Zadeh, L., *Fuzzy Logic and the Calculus of Fuzzy If--Then Rules*, OMRON Technics, Vol. 31, No. 4, 316-320, 1991.

Custódio, L. M. M., *Fundamentos de Inteligência Artificial – Acetatos*, Instituto Superior Técnico.

Cardoso, F., Fontes, F., Pais, C., Oliveira, P. and Ribeiro, M. I., *Fuzzy Logic Steering Controller For a Guided Vehicle, Proceedings of the* MELECON-94, *Mediterranean Electrotechnical Conference*, 1994, Antalya, Turkey.

von Altrock, C., *Fuzzy Logic and NeuroFuzzy Applications Explained*, Prentice Hall, Englewood Cliffs, NJ, 1995.

Bezdek, J. C., *Fuzzy Models --- What are they, and why?*, IEEE Transactions on Fuzzy Systems, Vol 1, 1-5, 1993.

Cox, Earl, *The Seven Nouble Truths of Fuzzy Logic*, Computer Design, April 1992.

Nomadic Technologies, *Nomad User's Manual* 1997.

The MathWorks, *MatLab 6.5 Help Desk*, 2002.

http://www.seattlerobotics.org/encoder/mar98/fuz/flindex.html

http://www.aptronix.com/fuzzynet/index.htm

http://www.emsl.pnl.gov:2080/proj/neuron/fuzzy/what.html